DOI:
10.5555/3091125.3091259
10.5555/3091125.3091259

Document Version
Peer reviewed version

Planning for Persuasion

Elizabeth Black
King's College London
elizabeth.black@kcl.ac.uk

Amanda J. Coles
King's College London
amanda.coles@kcl.ac.uk

Christopher Hampson
King's College London
christopher.hampson@kcl.ac.uk

ABSTRACT

We aim to find a winning strategy that determines the arguments a proponent should assert during a dialogue such that it will successfully persuade its opponent of some goal arguments, regardless of the strategy employed by the opponent. By restricting the strategies we consider for the proponent to what we call *simple strategies* and by modelling this as a planning problem, we are able to use an automated planner to generate *optimal simple strategies* for realistically sized problems. These strategies guarantee with a certain probability (determined by the proponent's uncertain model of the arguments available to the opponent) that the proponent will be successful no matter which arguments the opponent chooses to assert. Our model accounts for the possibility that the proponent, when asserting arguments, may give away knowledge that the opponent can subsequently use against it; we examine how this affects both time taken to find an optimal simple strategy and its probability of guaranteed success.

Keywords

argumentation, dialogue, persuasion, strategy, planning

1. INTRODUCTION

Argumentation theory allows reasoning with inconsistent and uncertain information and is a key sub-field of artificial intelligence [2], due in part to its potential to act as a bridge between human and machine reasoning [31]. Argument *dialogues* provide a principled way of structuring rational interactions between agents (be they human or machine) who argue about the validity of certain claims and each aim for a dialogue outcome that achieves their dialogue goal (*e.g.*, to persuade the other participant to accept their point of view [34], to reach agreement on an action to perform [3], or to persuade a human user to change their behaviour [23, 39]). Achievement of an agent's dialogue goal typically depends on both the arguments that the agent chooses to make during the dialogue, determined by its *strategy*, and the arguments asserted by its interlocutor. The strategising agent, which we refer to as the *proponent*, thus has the difficult problem of having to consider not only which arguments to assert but also the possible responses of its *opponent*. Since the opponent may make use of knowledge from arguments asserted by the proponent to construct new arguments, the proponent must also take care not to divulge information that its opponent can use against it. Recent works have considered how the proponent might use an uncertain (probabilistic) model of its opponent in order to guide its choice of which arguments to assert (*e.g.*, [4, 18, 22, 24, 25, 26, 36, 39]).

We consider a strategic argumentation setting where two agents exchange arguments, and the proponent aims to *persuade* its opponent of some goal arguments. Our proponent has an uncertain model of the opponent's arguments but no knowledge of its strategy, and so we aim to find a strategy that will be successful *no matter which arguments the opponent asserts*. This problem is PSPACE-complete [29] however by restricting the proponent strategies we consider to what we call *simple strategies* and translating our problem to a planning problem, we can use an automated planner to find strategies that have a certain probability of being effective, regardless of the strategy employed by the opponent.

The planning problem involves finding a sequence of permissible actions that will transform a given initial state into a state that satisfies the goal. In general, propositional planning is PSPACE-hard [8] however decades of research has led to scalable planning systems capable of solving realistic problems. Our contribution is to translate our strategic argumentation problem to a planning problem in the standard planning language PDDL2.1 [16] so that a solution to the problem (which we find using the POPF planner [13]) has a certain probability of being a winning strategy for the proponent no matter which arguments the opponent asserts. By repeatedly solving the proposed planning problem, each time updating the target probability to be greater than that of the previous solution, we can maximise the probability of guaranteed success and generate an optimal simple strategy. This does not necessarily imply that a solution found by our approach is an optimal strategy in general, since (to improve scalability) we restrict the planner to only search for simple strategies, which predetermine a sequence of arguments for the proponent to assert (in contrast, *e.g.*, to a policy where the proponent's assertions can depend on those of its opponent); however our results show that the solutions found by our approach typically perform well, with reasonable probability of guaranteed success.

The important challenge of how to generate proponent strategies for persuasion has not been widely explored [41]. Hadoux et al. [18], Rosenfeld and Kraus [39], Hadoux and Hunter [20] and Rienstra et al. [36], respectively, employ mixed observability Markov decision processes [32], partially observable Markov decision processes [28], decision trees, and a variant of the maxmin algorithm [11], to determine an effective proponent strategy; in contrast to our approach, none of these works aims to find a strategy that guarantees a certain probability of success no matter which arguments the opponent chooses to assert. Black et al. [4] and Hunter [22, 24, 25] also consider how a model of the arguments either known [4] or believed [22, 24, 25] by the opponent can be

Appears in: *Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017), S. Das, E. Durfee, K. Larson, M. Winikoff (eds.), May 8–12, 2017, São Paulo, Brazil.*

used to determine optimal dialogues, but for an asymmetric persuasion setting, where only the proponent may assert arguments.

Other related works take a game-theoretic approach to investigate strategic aspects of argumentation (*e.g.*, [27, 30, 35, 37]), but these typically depend on complete knowledge and the assumption that each participant will play optimally to maximise their utility. These assumptions are too strong for many settings (particularly if we are engaged in agent-human persuasion, as recent work has shown that humans do not typically conform to optimal strategies in an argumentative context [38]). We focus here on settings where the proponent may be unaware of the opponent's goal(s) and has no information about the opponent's strategy; we aim to find a strategy for the proponent that has a reasonable chance of being effective no matter which arguments the opponent chooses to make.

Hunter and Thimm [26] introduce the notion of *argumentation lotteries*, which can be used to capture uncertainty over the arguments and attacks known to an opponent and thus to determine the expected utility of a particular assertion with regards to whether it will bring about a desired dialectical outcome. In contrast, our approach considers the outcome that is determined from the eventual result of the dialogue, taking into account all possible assertions that the opponent may make in order to find an effective strategy.

Caminada [10] presents an argument dialogue system that can be used to explain why an argument is justified under the grounded semantics [14] and shows that this is sound and complete with respect to existence of a winning strategy for the proponent (under the assumption that the arguments available to the opponent are precisely known to the proponent). Our approach, in contrast, is applicable to any argumentation semantics, deals with uncertainty over the opponent's arguments, and aims to persuade the opponent no matter whether the goal argument(s) are justified by the chosen semantics.

We believe this is the first work to present an automated planning approach for producing effective strategies for symmetric persuasion that accounts for the uncertainty of the proponent's model of the opponent by finding strategies that have a certain probability of guaranteed success no matter which arguments the opponent chooses to assert. Furthermore, we believe the work presented here is the first to generate proponent strategies that account for the possibility that the opponent may exploit information obtained from the arguments asserted by the proponent to construct arguments unknown to it at the start of the dialogue.

This work expands a previous extended abstract [5], which gives an overview of our approach but no results. The remainder of the paper is structured as follows: In Section 2 we define the strategic argumentation problem addressed here. Section 3 introduces some formal preliminaries of planning with propositional and numerical variables, while in Section 4 we define the translation of our strategic argumentation problem to a planning problem. Section 5 investigates the performance of our approach, considering both the time taken to find an optimal simple strategy and its probability of success. We conclude with a discussion in Section 6.

2. STRATEGIC ARGUMENTATION

We focus on dialogues in which the *proponent* $\mathcal{P}$ and *opponent* $\mathcal{O}$ exchange sets of arguments. The proponent aims to *persuade* its opponent of the acceptability of some set of *goal arguments* $\mathcal{G}$. We make no assumptions about the opponent's dialogue goal. The arguments exchanged during a dialogue and the conflicts between these arguments can be represented as an *argumentation framework* [14].

DEFINITION 1. *An* argumentation framework *is a pair* $\mathsf{AF} = (\mathcal{A}, \rightarrow)$*, where* $\mathcal{A}$ *is a non-empty set of* arguments *and* $\rightarrow \subseteq (\mathcal{A} \times \mathcal{A})$ *is a binary* attack relation *on* $\mathcal{A}$*.*

Different semantics, capturing different rationality criteria, can be applied to an argumentation framework in order to determine subsets of arguments (called *extensions*) that can collectively be considered acceptable, given the conflicts present in the argumentation framework [14]. We define here particularly the *grounded semantics*, which we use in our evaluation (Section 5).

DEFINITION 2. *A* semantics *is a function* η *that is applied to an argumentation framework* $\mathsf{AF} = (\mathcal{A}, \rightarrow)$ *and returns a set of* extensions $\eta(\mathsf{AF}) \subseteq 2^{\mathcal{A}}$*. An extension* $E \in \eta(\mathsf{AF})$ *is* conflict-free *if there are no* $x, y \in E$ *such that* $x \rightarrow y$*. An extension* $E \subseteq \mathcal{A}$ defends *an argument* $x \in \mathcal{A}$ *if for every* $y \in \mathcal{A}$ *such that* $y \rightarrow x$*, there is some* $z \in E$ *such that* $z \rightarrow y$*. Let* E^D *denote that set of arguments defended by* E*. The* grounded *semantics* $gr(\mathsf{AF})$ *returns the (unique) subset-minimal conflict-free extension* E *of* AF *such that* $E = E^D$*.*

Using some chosen semantics, we can identify those arguments deemed to be *acceptable* with respect to a given framework. In the evaluation we present in Section 5, we use the *grounded acceptability function* (defined below) however our translation to a planning problem does not depend on the grounded semantics. Our approach requires only that some acceptability function is defined; one can instantiate this with either a credulous or sceptical attitude (see below) and with one's chosen semantics as desired.

DEFINITION 3. *Let* $\mathsf{acc}^{\theta}_{\eta}$ *be an* acceptability function *that assigns to each argumentation framework* AF *a set of* acceptable *arguments with respect to some semantics* η *and either a* credulous *(*$\theta = c$*) or* sceptical *(*$\theta = s$*) attitude as follows:* $\mathsf{acc}^{c}_{\eta} = \{a : a \in \bigcup_{E \in \eta(\mathsf{AF})} E\}$*,* $\mathsf{acc}^{s}_{\eta} = \{a : a \in \bigcap_{E \in \eta(\mathsf{AF})} E\}$*. Note that since the grounded semantics returns a unique extension, we denote the* grounded acceptability function *as* $\mathsf{acc}_{gr}(\mathsf{AF}) = \{a : a \in gr(\mathsf{AF})\}$*.*

Agents are represented by arguments available to them at the start of the dialogue, together with a closure operator that is used to *'derive'* new arguments from a given set of arguments. This allows us to handle the case where the opponent can construct new arguments unknown to them at the start of the dialogue by combining constituent elements of arguments asserted by the proponent with its own knowledge. This same feature is achieved in [40] by means of a set of 'derivation rules' of the form $a_0, \ldots, a_k \Rightarrow b$, where $a_0, \ldots, a_k, b$ are arguments. We assume agents share the same attack relation and so represent the set of all arguments that can be available to either agent by the argumentation framework $\mathsf{AF} = (\mathcal{A}, \rightarrow)$.

DEFINITION 4. *An* agent model *is a tuple* $\mathsf{Ag} = (\mathcal{K}_{\mathsf{Ag}}, \mu_{\mathsf{Ag}})$*, where* $\mathcal{K}_{\mathsf{Ag}} \subseteq \mathcal{A}$ *is the set of arguments available to the agent, and* $\mu_{\mathsf{Ag}} : 2^{\mathcal{A}} \rightarrow 2^{\mathcal{A}}$ *describes a* closure operator *on* $\mathcal{A}$ *such that, for all* $\mathcal{B}, \mathcal{C} \subseteq \mathcal{A}$*, the following conditions hold:*

- (extensivity) $\mathcal{B} \subseteq \mu_{\mathsf{Ag}}(\mathcal{B})$*,*
- (monotonicity) *if* $\mathcal{B} \subseteq \mathcal{C}$ *then* $\mu_{\mathsf{Ag}}(\mathcal{B}) \subseteq \mu_{\mathsf{Ag}}(\mathcal{C})$*,*
- (idempotence) $\mu_{\mathsf{Ag}}(\mu_{\mathsf{Ag}}(\mathcal{B})) = \mu_{\mathsf{Ag}}(\mathcal{B})$*.*

Our proponent has an *uncertain* model of its opponent, described by a probability distribution over a set of possible agent models.

DEFINITION 5. *An* (uncertain) opponent model *is a pair* $\mathcal{M} = (\mathcal{E}, p)$*, where* $\mathcal{E}$ *is a finite set of agent models (which we refer to as* possible opponent models*) and* $p : \mathcal{E} \to \mathbb{Q}$ *is a probability distribution over* $\mathcal{E}$ *such that* $p(\mathcal{O}_i) > 0$*, for each* $\mathcal{O}_i \in \mathcal{E}$*, and* $\sum_{\mathcal{O}_i \in \mathcal{E}} p(\mathcal{O}_i) = 1$*, where* $p(\mathcal{O}_i)$ *denotes the proponent's perceived likelihood that its opponent can be represented by the agent model* $\mathcal{O}_i$*.*

We consider dialogues where the agents take turns to assert sets of arguments, not repeating arguments previously asserted, and that terminate only when each has nothing further they wish to assert. A dialogue is *successful* for the proponent if its goal arguments are determined to be acceptable under the chosen acceptability function, given the arguments asserted during the dialogue.

DEFINITION 6. *We define a* dialogue *to be a sequence of* moves $D = [M_0, M_1, \ldots, M_n]$ *where each move* $M_k \subseteq \mathcal{A}$ *is a finite set of arguments, such that* $M_i \cap M_j = \emptyset$*, for* $i \neq j$*, and* $M_{k-1} \cup M_k \neq \emptyset$*, for* $k < n$*. Let* $\mathsf{args}(D) = \bigcup_{k \leq n} M_k$ *denote the set of arguments asserted during the dialogue. A dialogue is* terminated *iff* $M_n = M_{n-1} = \emptyset$*. A dialogue* D *is* successful *with respect to a set of goal arguments* $\mathcal{G} \subseteq \mathcal{A}$ *iff* $\mathcal{G} \subseteq \mathsf{acc}^{\theta}_{\eta}(\mathsf{AF}_D)$*, where* $\mathsf{acc}^{\theta}_{\eta}$ *is the chosen acceptability function and* $\mathsf{AF}_D = (\mathsf{args}(D), \rightarrow_D)$ *is the argumentation framework induced by* D *on* AF*, where* $\rightarrow_D = \rightarrow \cap (\mathsf{args}(D) \times \mathsf{args}(D))$.

Note that we do not aim here to model human persuasion, which may involve richer interactions than the simple exchange of arguments. Nevertheless, understanding how an agent can select which arguments to assert in a strategic argumentation setting such as we define here is an important and timely challenge (see, *e.g.*, [18, 20, 29, 36, 39, 41]) key to understanding how an agent can select persuasive arguments. Furthermore, recent works have shown how dialogues in which a proponent agent is able only to assert arguments can be applied in order to bring about behaviour change in a human user [24, 39].

An agent's *strategy* determines the moves it makes during a dialogue. We assume that agents only assert arguments from their private knowledge $\mathcal{K}_{\mathsf{Ag}}$ together with any inferred arguments that they can derive (with their closure operator μ_{Ag}) from their private knowledge and the arguments asserted thus far. As the success of a dialogue is determined only by the arguments that have been asserted and not the order in which they have been asserted, we consider strategies that, likewise, consider only those arguments that have been asserted, regardless of their position in the dialogue.

DEFINITION 7. *A* (general) strategy *for* $\mathsf{Ag} = (\mathcal{K}_{\mathsf{Ag}}, \mu_{\mathsf{Ag}})$ *is a function* $\sigma_{\mathsf{Ag}} : 2^{\mathcal{A}} \to 2^{\mathcal{A}}$ *such that* $\sigma_{\mathsf{Ag}}(\mathcal{B}) \subseteq \mu_{\mathsf{Ag}}(\mathcal{K}_{\mathsf{Ag}} \cup \mathcal{B})$ *and* $\sigma_{\mathsf{Ag}}(\mathcal{B}) \cap \mathcal{B} = \emptyset$*, for all* $\mathcal{B} \subseteq \mathcal{A}$*. Given a pair of strategies* $(\sigma_{\mathcal{P}}, \sigma_{\mathcal{O}})$*, let* $D_{(\sigma_{\mathcal{P}}, \sigma_{\mathcal{O}})} = [M_0, \ldots, M_n]$ *be the dialogue such that* $M_k = \sigma_{\mathsf{Ag}}(M_0 \cup \cdots \cup M_{k-1})$*, for all* $k \leq n$*, where* $\mathsf{Ag} = \mathcal{P}$ *whenever* k *is even and* $\mathsf{Ag} = \mathcal{O}$ *whenever* k *is odd.*

We aim to find a strategy that will lead to a successful dialogue no matter which possible opponent model is accurate and regardless of the opponent's strategy. As it may be the case that no such strategy exists, we consider instead strategies with greater than λ probability of being a winning strategy, with respect to the proponent's model of the opponent.

DEFINITION 8. *A strategy* $\sigma_{\mathcal{P}}$ *is* effective *against an opponent* $\mathcal{O}$ *if there is no strategy* $\sigma_{\mathcal{O}}$ *that* $\mathcal{O}$ *could play such that* $D_{(\sigma_{\mathcal{P}}, \sigma_{\mathcal{O}})}$ *is terminated but unsuccessful. Let* $(\mathcal{E}, p)$ *be* $\mathcal{P}$*'s opponent model and* $\mathcal{E}' \subseteq \mathcal{E}$ *be the set of possible opponent models against which* $\sigma_{\mathcal{P}}$ *is effective, then* $\sigma_{\mathcal{P}}$ *is a* λ-winning strategy *iff* $\sum_{\mathcal{O}_i \in \mathcal{E}'} p(\mathcal{O}_i) > \lambda$*.*

$\lvert\mathcal{A}\rvert$	*All strategies*	*All simple strategies*
4	4.29×10^9	26
6	6.28×10^{57}	1,081
8	1.80×10^{308}	94,586

Table 1: Number of possible simple strategies compared with the total number of possible strategies.

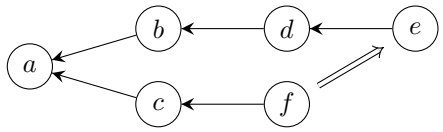

Figure 1: Illustration of AF and μ from Example 1

Our **strategic argumentation problem** is as follows:

Instance: Given an agent model $\mathcal{P}$ together with an opponent model $\mathcal{M}$, and a target probability $\lambda \in [0, 1]$,

Problem: Find a λ-winning strategy for $\mathcal{P}$, with respect to $\mathcal{M}$.

For examples with more than 6 arguments, the number of possible general strategies exceeds the number of atoms in the known universe (estimated to be between 10^{78} and 10^{82}) (Table 1). To improve scalability of our approach, we restrict the strategies we consider to *simple strategies* that can be specified by a finite sequence of (non-intersecting) moves $S = [S_0, \ldots, S_n]$, such that $S_i \subseteq \mathcal{K}_{\mathcal{P}}$ for $i \leq n$. These moves are followed sequentially by the proponent unless the dialogue thus far is successful, in which case the proponent chooses not to assert any arguments (asserting $\emptyset$). If the opponent also chooses not to assert any arguments at this point then the dialogue will terminate; otherwise if the opponent makes an assertion that causes the dialogue to change to unsuccessful, the proponent will continue with the next unasserted move in its sequence S. If the proponent exhausts its sequence of moves S then it will respond with $\emptyset$ until the responder also asserts $\emptyset$, thereby terminating the dialogue. More formally, the *simple strategy* $\sigma^S_{\mathcal{P}}$, associated with S, is defined such that for every dialogue $D = [M_0, \ldots, M_{2k-1}]$,

$$\sigma^S_{\mathcal{P}}(D) = \begin{cases} \emptyset & \text{if } D \text{ is successful or } j > n \\ S_j & \text{otherwise.} \end{cases}$$

where $j = |\{i < k : M_{2i} \neq \emptyset\}|$ (*i.e.*, S_j is the earliest element of S that $\mathcal{P}$ has not yet asserted in dialogue D).

Simple strategies form a proper subclass of general strategies, since they cannot capture more elaborate policies that depend on previous moves. Our approach must, therefore, take into account all possible dialogues that may arise given the proponent's opponent model, in order to find a simple strategy that maximises λ, the probability of guaranteed success no matter what strategy the opponent employs. While success of a specific dialogue does not depend on the order arguments are asserted in, the order and grouping of arguments to assert *does* affect the effectiveness of simple strategies (as we see in Example 1) since different arguments may be effective against different possible opponent models.

EXAMPLE 1. *Let* $\mathsf{AF} = (\mathcal{A}, \rightarrow)$ *be the argumentation framework illustrated in Figure 1, where* $\mathcal{A} = \{a, b, c, d, e, f\}$ *and* $\rightarrow = \{(e, d), (d, b), (b, a), (c, a), (f, c)\}$*, and let* $\mu : 2^{\mathcal{A}} \to 2^{\mathcal{A}}$ *be a closure operator such that for all* $\mathcal{B} \subseteq \mathcal{A}$*, if* $f \in \mathcal{B}$ *then* $\mu(\mathcal{B}) =$

$\mathcal{B} \cup \{e\}$, *otherwise* $\mu(\mathcal{B}) = \mathcal{B}$ *(i.e., the closure operator generated by the inference rule* $f \Rightarrow e$*). Consider the following sets of arguments:*

$$\mathcal{K}_0 = \{b\}, \qquad \mathcal{K}_1 = \{c\}, \qquad \mathcal{K}_2 = \{b, c\}.$$

Suppose $\mathcal{P} = (\mathcal{A}, \mu)$ *is a proponent with goal* $\mathcal{G} = \{a\}$*, and is using the grounded acceptability function. Let* $(\mathcal{E}, p)$ *be an opponent model, where* $\mathcal{E}$ *comprises three possible models* $\mathcal{O}_i = (\mathcal{K}_i, \mu)$*, for* $i < 3$*, and* $p : \mathcal{E} \to \mathbb{Q}$ *is such that* $p(\mathcal{O}_0) = 0.4$*,* $p(\mathcal{O}_1) = 0.5$ *and* $p(\mathcal{O}_2) = 0.1$*.*

The simple strategy $S = [\{a, d\}]$ *is effective against* $\mathcal{O}_0$*, but not against* $\mathcal{O}_1$ *or* $\mathcal{O}_2$*, and so is a* 0.4*-winning strategy. The simple strategy* $S' = [\{a, d, f\}]$ *is a* 0.5*-winning strategy, as it is effective against* $\mathcal{O}_1$*, but not against* $\mathcal{O}_0$ *because the argument* e*, unknown initially to* $\mathcal{O}_0$*, can be inferred from* f*. The simple strategy* $S'' = [\{a, d\}, \{f\}]$ *is a* 0.9*-winning strategy, failing only in the case that the opponent knows both* b *and* c*.*

3. NUMERIC PLANNING

We consider the classical planning problem with conditional effects and bounded numerical variables. Here we define a planning problem over the subset of the Planning Domain Definition Language PDDL2.1 [16] needed to support our translation.

Let Σ_P be a finite set of propositional variables and Σ_N be a finite set of numerical variables, such that $\Sigma_P \cap \Sigma_N = \emptyset$. A *propositional condition* takes the form $(P = t)$, while a *propositional effect* takes the form $(P \leftarrow t)$, for $P \in \Sigma_P$ and $t \in \{\top, \bot\}$. For readability, we abbreviate both propositional conditions and effects with a single literal P if $t = \top$, or $\neg P$ if $t = \bot$. A *numerical condition* takes the form $(V \bowtie q)$, while a *numerical effect* takes the form $(V \leftarrow q)$ or $(V \leftarrow V + q)$, for $V \in \Sigma_N$, $q \in \mathbb{Q}$ and $\bowtie \in \{=, <, >\}$.

Our *state space* Ω comprises a set of functions that assign a value $t \in \{\top, \bot\}$ to each $P \in \Sigma_P$ and a value $q \in \mathbb{Q}$ to each $V \in \Sigma_N$. A state $\omega \in \Omega$ satisfies a propositional condition $(P = t)$ iff $\omega(P) = t$, and a numerical condition $(V \bowtie q)$ iff $\omega(V) \bowtie q$. We write $\omega \models L$ when ω satisfies the condition L, and $\omega \models (L \vee L')$ iff $\omega \models L$ or $\omega \models L'$. For any set of conditions C, we write $\omega \models C$ iff $\omega \models L$, for all $L \in C$.

DEFINITION 9. *A* planning problem *is specified by a tuple* $P = \langle \Sigma_P, \Sigma_N, I, G, A \rangle$*, where:*

- Σ_P *is a set of propositional variables,*
- Σ_N *is a set of numerical variables,*
- I *is a finite set of* initial *conditions,*
- G *is a finite set of* goal *conditions,*
- A *is a set of* actions *of the form* $\alpha = \langle pre(\alpha), \mathit{eff}(\alpha) \rangle$*, where* $pre(\alpha)$ *is a set of* (pre-)conditions *and* $\mathit{eff}(\alpha)$ *is a set of* conditional effects *of the form* $(\textbf{if } C \textbf{ then } L)$*, where* C *is a (possibly empty) set of conditions, and* L *is an effect.*

The transition function $\delta : \Omega \times A \to \Omega$ is such that $\delta(\omega, \alpha) = \omega'$ iff (i) $\omega \models pre(\alpha)$, (ii) if $\omega \models C$ then $\omega'(L) = t$, for all $(\textbf{if } C \textbf{ then } (L \leftarrow t)) \in \mathit{eff}(\alpha)$, and (iii) $\omega'(L) = \omega(L)$ if L does not occur in any effect of $\mathit{eff}(\alpha)$. We extend this transition function to sequences of actions, recursively by taking $\delta^*(\omega, [\,]) = \omega$ and $\delta^*(\omega, [\alpha_0, \ldots, \alpha_k]) = \delta(\delta^*(\omega, [\alpha_0, \ldots, \alpha_{k-1}]), \alpha_k)$, for all $\omega \in \Omega$ and $\alpha_0, \ldots, \alpha_k \in A$. The planning problem is *well-defined* whenever δ is a well-defined function.

DEFINITION 10. *A* solution *to a well-defined planning problem* P *is a sequence of actions* $\vec{\alpha} = [\alpha_0, \ldots, \alpha_n]$ *such that* $\delta^*(\omega_I, \vec{\alpha}) \models G$*, whenever* $\omega_I \models I$*.*

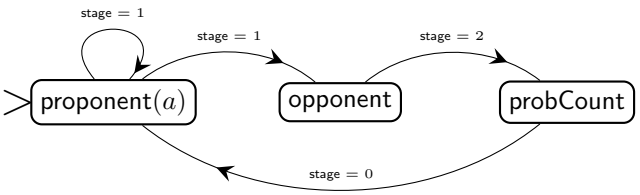

Figure 2: Order in which actions may be applied

4. MODELLING STRATEGIC ARGUMENTATION AS A PLANNING PROBLEM

We show how our strategic argumentation problem can be modelled as a planning problem with conditional effects and bounded numerical variables. The major challenge here is that the initial state is unknown (since the proponent does not know which of the possible opponent models holds true) and the possible actions are non-deterministic (since the proponent cannot know which moves the opponent will make). We would like to find a *conformant* plan, *i.e.*, one that will be successful no matter how the uncertainty in the problem is resolved.

The state-of-the-art for solving conformant planning problems is to compile away the uncertainty inherent in the problem so that it can be solved by a planner that assumes complete knowledge and deterministic actions [1]. Following this approach, we push the uncertainty about the opponent's arguments and strategy into the state description, so that, for each possible opponent model, we keep track of every possible dialogue that could arise given the proponent's moves. We determine the probability of success λ for a particular plan by considering all possible dialogues associated with each possible opponent model; only if all such dialogues are successful is the plan guaranteed to be effective against that possible opponent model, regardless of how the opponent chooses to play. We sum the probability of each such model to give λ.

Our translation comprises three types of actions. For each argument a known to the proponent, there is a $(\mathsf{proponent}(a))$ action, which emulates the act of the proponent asserting a. A single move is built up by iterated application of these actions (simulating the assertion of a set of arguments). The opponent's move is captured by a single $(\mathsf{opponent})$ action, which simulates *all* possible responses for each possible opponent model, adding them to a 'pool' of possible dialogue states associated with that model. Finally the $(\mathsf{probCount})$ action is applied after each $(\mathsf{opponent})$ action and sums the total probability of guaranteed success, against each of the possible opponent models $\mathcal{O}_i \in \mathcal{E}$. We regulate the order that different types of actions may appear in the plan with a numerical variable (stage), whose value determines which preconditions are satisfied and is updated by the effects of each action. This ordering is illustrated in Figure 2. We now formally define our translation.

Let $\mathcal{P} = (\mathcal{K}_\mathcal{P}, \mu_\mathcal{P})$ be our proponent model, with goal arguments $\mathcal{G} \subseteq \mathcal{A}$, uncertain opponent model $\mathcal{M} = (\mathcal{E}, p)$, and target probability $\lambda \in [0, 1]$. In what follows, let $\mathcal{A}_\mathcal{P} = \mu_\mathcal{P}(\mathcal{K}_\mathcal{P})$ be the set of arguments available to $\mathcal{P}$, and $\mathcal{A}_\mathcal{O} = \bigcup_{\mathcal{O}_i \in \mathcal{E}} \mu_{\mathcal{O}_i}(\mathcal{K}_{\mathcal{O}_i} \cup \mathcal{A}_\mathcal{P})$ be the set of all arguments belonging to any $\mathcal{O}_i \in \mathcal{E}$, together with any arguments that can be inferred with the help of any arguments from $\mathcal{A}_\mathcal{P}$ (*i.e.*, $\mathcal{A}_\mathcal{O}$ comprises all arguments that may be available to any possible opponent model throughout a dialogue). The planning problem $P^\lambda_{(\mathcal{P},\mathcal{M})}$ is then described as follows.

Variables We require the following numerical variables: (stage), $(\mathsf{probSuccess})$, and $(\mathsf{prob}(i))$, for each $\mathcal{O}_i \in \mathcal{E}$. The variable (stage), as described above, regulates the order in which actions

may appear in the plan, while (probSuccess) and (prob(i)) are used to calculate the probability of success for a given plan. In addition to these numerical variables we have the following propositional variables:

- canAssertP(a) means $\mathcal{P}$ has not yet asserted $a \in \mathcal{A}_\mathcal{P}$,
- canAssertO(M, i, D) says that $M \subseteq \mathcal{A}_\mathcal{O}$ is a possible move that can be played by $\mathcal{O}_i \in \mathcal{E}$, given that $\mathcal{P}$ has asserted $D \subseteq \mathcal{A}_\mathcal{P}$ (and so M consists of arguments that are either known to $\mathcal{O}_i$ or that it can infer from the arguments in D using its closure operator μ_i),
- dialogueP(D) says that $D \subseteq \mathcal{A}_\mathcal{P}$ is the set of arguments that have been asserted by $\mathcal{P}$,
- dialogueO(i, D) says that $D \subseteq \mathcal{A}_\mathcal{O}$ could have been asserted by $\mathcal{O}_i \in \mathcal{E}$ (and so keeps track of all possible dialogues that could arise for each possible opponent model),
- temp(i, D) acts as a temporary 'storage' variable for capturing dialogueO(i, D),
- successful$(D_\mathcal{P}, D_\mathcal{O})$ says that $D_\mathcal{P} \cup D_\mathcal{O}$ determines a successful dialogue, where $D_\mathcal{P} \subseteq \mathcal{A}_\mathcal{P}$ comprises the arguments asserted by the proponent, and $D_\mathcal{O} \subseteq \mathcal{A}_\mathcal{O}$ comprises arguments that may be asserted by the opponent,
- effective(i) means the plan is effective against $\mathcal{O}_i \in \mathcal{E}$,
- addP(a, D, D') says that D' is the result of adding a single argument $a \in \mathcal{A}_\mathcal{P}$ to D (*i.e.*, $D' = D \cup \{a\}$),
- addO(M, D, D') says that D' is the result of adding the *set* M to D (*i.e.*, $D' = D \cup M$).

The reason for the discrepancy between addP and addO is that, for the purposes of optimisation, we recursively add proponent arguments one at a time, while opponent arguments must be considered as a single move.

Initial Conditions Our set of initial conditions I comprises the following numerical conditions:

$$(\text{stage} = 0), \quad (\text{probSuccess} = 0), \quad (\text{prob}(i) = p(\mathcal{O}_i))$$

for all $\mathcal{O}_i \in \mathcal{E}$, where $p(\mathcal{O}_i) \in \mathbb{Q}$ is the probability of $\mathcal{O}_i$. We also require the following propositional initial conditions:

- canAssertP(a), for all $a \in \mathcal{A}_\mathcal{P}$,
- dialogueP$(\emptyset)$ and dialogueO$(i, \emptyset)$, for all $\mathcal{O}_i \in \mathcal{E}$,

together with the following conditions that, once set, remain unaltered by the effects of any of the actions:

- canAssertO$(M, i, D_\mathcal{P})$ iff $M \subseteq \mu_{\mathcal{O}_i}(\mathcal{K}_{\mathcal{O}_i} \cup D_\mathcal{P})$,
- successful$(D_\mathcal{P}, D_\mathcal{O})$ iff $\mathcal{G} \subseteq \text{acc}^\theta_\eta(D_\mathcal{P} \cup D_\mathcal{O})$,
- addP$(a, D_\mathcal{P}, D'_\mathcal{P})$ iff $D'_\mathcal{P} = D_\mathcal{P} \cup \{a\}$,
- addO$(M, D_\mathcal{O}, D'_\mathcal{O})$ iff $D'_\mathcal{O} = D_\mathcal{O} \cup M$,

where acc^θ_η is the agreed acceptability function for AF (and so our translation is applicable to any instantiation of acc^θ_η).

Goal Conditions The planning goal G comprises a single condition (probSuccess $> \lambda$).

Actions Our set of actions A comprises the three types of action discussed above, whose preconditions and effects are described in Tables 2–4, where all free parameters are universally quantified. So, for example, the proponent(a) action contains an effect of the form (**if** dialogueO$(i, D_\mathcal{O})$ **then** temp$(i, D_\mathcal{O})$, ¬dialogueO$(i, D_\mathcal{O})$), for all $\mathcal{O}_i \in \mathcal{E}$ and $D_\mathcal{O} \subseteq \mathcal{A}_\mathcal{O}$.

Action: proponent(a)	for all $a \in \mathcal{A}_\mathcal{P}$
pre	(i) canAssertP(a) (ii) (stage $= 0$) $\vee$ (stage $= 1$)
eff	(a) ¬canAssertP(a) (b) (stage $\leftarrow 1$) (c) **if** dialogueP$(D_\mathcal{P})$, addP$(a, D_\mathcal{P}, D'_\mathcal{P})$ **then** dialogueP$(D'_\mathcal{P})$, ¬dialogueP$(D_\mathcal{P})$ (d) **if** dialogueO$(i, D_\mathcal{O})$ **then** temp$(i, D_\mathcal{O})$, ¬dialogueO$(i, D_\mathcal{O})$ (e) effective(i)

Table 2: Preconditions and effects for (proponent(a))

Action: opponent	
pre	(i) (stage $= 1$)
eff	(a) (stage $\leftarrow 2$) (b) **if** dialogueP$(D_\mathcal{P})$, temp$(i, D_\mathcal{O})$, canAssertO$(M, i, D_\mathcal{P})$, addO$(M, D_\mathcal{O}, D'_\mathcal{O})$, ¬successful$(D_\mathcal{P}, D'_\mathcal{O})$ **then** dialogueO$(i, D'_\mathcal{O})$, ¬effective(i) (c) ¬temp$(i, D_\mathcal{O})$

Table 3: Preconditions and effects for (opponent)

For each $a \in \mathcal{A}_\mathcal{P}$ there is an action (proponent(a)) with preconditions and effects defined in Table 2. The (proponent(a)) action can be applied if the canAssertP(a) proposition is true (Table 2:(i)), and the stage variable takes a value of 0 or 1 (Table 2:(ii)). These actions emulate the proponent asserting the argument a during their turn of the dialogue. Iterated application allows the proponent to assert a set of arguments in a move. The effects of (proponent(a)) include: negating canAssertP(a), preventing an argument from being asserted more than once (Table 2:(a)); updating the stage variable to 1 (Table 2:(b)); and updating the contents of the dialogueP variable with the argument a (Table 2:(c)). There is a 'book-keeping' effect which copies the contents of the dialogueO variable into a temporary variable temp (Table 2:(d)); this is to prevent conflicts between 'add' and 'delete' effects in the following action. Finally, effective(i) is set to true for each possible opponent model $\mathcal{O}_i \in \mathcal{E}$ (Table 2:(e)), indicating that the current plan is (potentially) effective against that model; when the (opponent) action is applied, if there are any dialogues that may have occurred against $\mathcal{O}_i$ such that $\mathcal{O}_i$ can make a move such that the dialogue is then unsuccessful for the proponent, then effective(i) will be set to false and the subsequent (probCount) action will not collect the probability of $\mathcal{O}_i$.

A single (opponent) action (Table 3) emulates the effect of the opponent's move in the dialogue. It achieves this by considering each possible set of arguments $D_\mathcal{P} \cup D_\mathcal{O}$ that could have been asserted thus far against each possible opponent model $\mathcal{O}_i$ (Table 3:(b):line 1), and considers all possible moves $M \subseteq \mathcal{A}_\mathcal{O}$ that $\mathcal{O}_i$ could assert, in this instance (taking into account the knowledge obtained from the closure operator) (Table 3:(b):line 2). If the result of adding M to the dialogue is advantageous to the opponent (Table 3:(b):line 3), we update the contents of the dialogueO variable from $D_\mathcal{O}$ to $D'_\mathcal{O} = D_\mathcal{O} \cup M$ and record this strategy as ineffective against that possible opponent model (Table 3:(b):line 4). Without loss of generality, we need not consider the case where the dialogue is successful for the proponent after the opponent's move, since the response of our simple strategy will then be to assert $\emptyset$ until the

Action: probCount	
pre	(i) (stage $= 2$)
eff	(a) (stage $\leftarrow 0$) (b) **if** effective(i) **then** (probSuccess $\leftarrow$ probSuccess + prob(i)), (prob(i) $\leftarrow 0$)

Table 4: Preconditions and effects for (probCount)

opponent chooses to assert an argument for which we must reply. As the opponent will not gain any new information from the proponent in this case, anything they may later chose to assert could have been asserted in their previous move. Since we are considering *all* possible dialogues that the proponent could encounter, this possibility is already considered. Another 'book-keeping' effect clears the contents of temp($i', D_{\mathcal{O}}$) (Table 3:(c)).

A single (probCount) action is applied after each (opponent) action (Table 4:(i)) and accumulates the probability of any possible opponent models against which the current plan is effective (Table 4:(b):lines 1–2). This is precisely those models for which there is no move that opponent can play that will make the dialogue unsuccessful. After the probability of a possible opponent model has been counted, it is set to zero to prevent double-counting (Table 4:(b):line 3).

It follows from these definitions that $P^{\lambda}_{(\mathcal{P},\mathcal{M})}$ has a solution if and only if there is a λ-winning simple strategy for $\mathcal{P}$, with respect to the opponent model $\mathcal{M}$. Moreover, one can read off such a strategy directly from the planning solution by collecting together all consecutive (proponent(a)) actions into a single move, with each move separated by the (opponent) and (probCount) actions. To find an *optimal* simple strategy that maximises λ, we first find a solution where $\lambda > 0$ and then iteratively seek solutions where λ is greater than the probability of success of the previous solution, until no such solution exists, guaranteeing the last solution found to be an optimal simple strategy.

5. EVALUATION

We automated the above translation and used an implementation of the POPF planner [13], which natively supports ADL (the Action Description Language, which provides universal and existential quantification [33]) and conditional effects, to generate optimal simple strategies for several examples, described below. Native support for ADL allows us to represent the problem more compactly and is why we use this planner, which is sound and complete, and applies a heuristic forward search. We were unable to use planners that ground out quantification before search (*e.g.*, Metric-FF [21]) as, for larger examples, this exhausts available memory.

We benchmark against a naive brute-force algorithm, that enumerates every possible simple strategy and calculates the probability of success for each. This involves a depth-first-search, for each possible opponent model $\mathcal{O}_i \in \mathcal{E}$, of all the possible dialogues that can be generated by $\mathcal{O}_i$, until an unsuccessful terminating dialogue is found. The probability of success is the sum of the probabilities $p(\mathcal{O}_i)$ for which no such dialogue can be found. If a strategy is found to be effective against all $\mathcal{O}_i \in \mathcal{E}$ then this strategy is optimal and we terminate the search, otherwise the naive algorithm must check every possible simple strategy to determine which are optimal. To ensure fairness of comparison with the times reported by the planner, we include only the search time and not the time taken to calculate whether a particular set of arguments is successful, which is supplied at run-time.

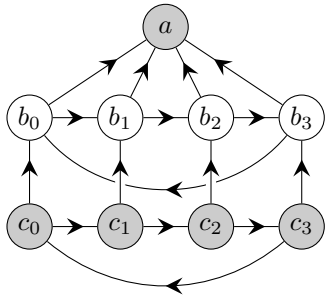

Figure 3: Illustration of $cycle_4$.

To compare the performance of the two approaches, we ran experiments on a selection of argumentation frameworks, using the grounded acceptability function to determine dialogue success. Note that the information about which sets of arguments determine the goal argument(s) to be acceptable is supplied at run time (with the successful predicate) and thus the search times we report here are unaffected by the complexity of determining this. The complexity of credulous and sceptical acceptance decision problems is well-understood (*e.g.*, [15]) and known to be intractable in the worst case for some of the standard semantics. Nevertheless, many advanced argument solvers have been developed (see [12] for a review) and shown to be capable of reasoning with large argumentation frameworks (*e.g.*, [42]). For the argumentation frameworks considered here, the pre-processing time required to set the initial conditions of the planning problem was negligible.

The argumentation frameworks we used in our evaluation include those described in [18], labelled as they are referred to in that paper: Ex 1, Dv., 7, 8, and 9. We do not have space to repeat the description of those frameworks here but note that their sizes range from 7 to 9 arguments and, excepting Dv., they are all bipartite and so the proponent need not worry about undermining their own arguments. In order to examine the performance of our approach in more diverse settings and with larger problems, we consider two other (non-bipartite) families of argumentation framework: one with *cycles* and one where arguments can be both attackers and supporters of the goal argument (referred to here as type *ladder*). We chose these framework types because of the challenges posed by the existence of arguments that may be either helpful or harmful for the proponent, depending on the opponent's actual arguments and strategy.

DEFINITION 11. *Examples labelled $cycle_n$ comprise of an argumentation framework with arguments $\mathcal{A} = \{a\} \cup \{b_i, c_i : i < n\}$ and attack relation $\rightarrow = \{(b_i, a), (c_i, b_i) : i < n\} \cup \{(b_{i-1}, b_i), (c_{i-1}, c_i) : 0 < i < n\} \cup \{(b_{n-1}, b_0), (c_{n-1}, c_0)\}$, such that goal arguments $\mathcal{G} = \{a\}$, $\mathcal{A}_{\mathcal{P}} = \{a\} \cup \{c_i : i < n\}$ and for all $(\mathcal{K}_{\mathcal{O}_i}, \mu_{\mathcal{O}_i}) \in \mathcal{E}$ (where $(\mathcal{E}, p)$ is the proponent's uncertain opponent model) $\mathcal{K}_{\mathcal{O}_i} \subseteq \{b_i : i < n\}$.*

Figure 3 shows an example of a cycle argumentation framework, $cycle_4$, where the grey arguments are those available to the proponent and the white arguments are those available to the opponent. We see here that, for example, asserting the argument c_0 could be beneficial to the proponent if the opponent knows the argument b_0 (as c_0 attacks b_0) but if the opponent knows b_1 then asserting c_0 could be detrimental to the proponent's goal (since c_0 attacks c_1, which is the only argument available that attacks b_1). The proponent must therefore take care to consider all the possible opponent models in order to determine whether an argument is likely to be detrimental to achieving its goal.

Example	Ex 1				Dv.				7				8				9				$cycle_4$			
size $\|\mathcal{E}\|$	1	2	4	8	1	2	4	8	1	2	4	8	1	2	4	8	1	2	4	8	1	2	4	8
Planner (s)	**0.03**	**0.07**	**0.17**	**0.36**	**0.03**	**0.07**	**0.16**	**0.36**	0.01	0.03	0.07	0.14	**0.03**	**0.06**	**0.16**	**0.42**	**0.06**	**0.22**	**0.69**	1.99	**0.1**	0.27	0.93	2.8
Naive (s)	0.97	1.75	3.75	7.33	0.12	0.26	0.50	1.04	**0.00**	**0.00**	**0.01**	**0.02**	0.18	0.34	0.71	1.40	0.19	0.47	0.88	**1.50**	0.11	**0.23**	**0.45**	**0.89**
Prob. λ	1.00	1.00	1.00	1.00	0.45	0.48	0.53	0.50	0.56	0.6	0.57	0.58	1.00	1.00	1.00	1.00	0.74	0.77	0.77	0.79	0.46	0.42	0.42	0.40

Example	$cycle_5$				$cycle_6$				$ladder_4$				$ladder_5$				$ladder_6$			
size $\|\mathcal{E}\|$	1	2	4	8	1	2	4	8	1	2	4	8	1	2	4	8	1	2	4	8
Planner (s)	**1.31**	6.08	24.8	55.3	**23.0**	**162.9**	**244.3**	**204.4**	**0.08**	**0.20**	**0.75**	**2.35**	**0.97**	**4.05**	**10.7**	**25.2**	**12.8**	**53.9**	**155.8**	**202.1**
Naive (s)	2.07	**4.04**	**8.37**	**16.7**	120.1	301.4	526.1	973.7	0.63	1.17	2.40	5.14	9.27	18.7	37.2	77.4	2547.7	3309.2	3595.1	3601.0
Prob. λ	0.3	0.34	0.33	0.33	0.28	0.23	0.28	0.23	0.71	0.79	0.74	0.76	0.56	0.62	0.61	0.64	0.79	0.72	0.75	0.76

Table 5: Average search times (secs.) to find an optimal simple strategy and average probability of guaranteed success

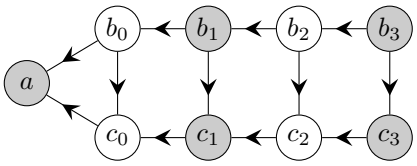

Figure 4: Illustration of $ladder_4$.

DEFINITION 12. *Examples labelled $ladder_n$ comprise of an argument framework with arguments $\mathcal{A} = \{a\} \cup \{b_i, c_i : i < n\}$ and attack relation $\rightarrow = \{(b_0, a), (c_0, a)\} \cup \{(b_i, b_{i-1}), (c_i, c_{i-1}) : 0 < i < n\} \cup \{(b_i, c_i) : i < n\}$, such that goal arguments $\mathcal{G} = \{a\}$, $\mathcal{A}_\mathcal{P} = \{a\} \cup \{b_i, c_i : i = 1, 3, \dots\}$ and for all $(\mathcal{K}_{\mathcal{O}_i}, \mu_{\mathcal{O}_i}) \in \mathcal{E}$ (where $(\mathcal{E}, p)$ is the proponent's uncertain opponent model) $\mathcal{K}_{\mathcal{O}_i} \subseteq \{b_i, c_i : i = 0, 2, \dots\}$.*

An example of a ladder argumentation framework, $ladder_4$, is shown in Figure 4, where the grey arguments are those available to the proponent and the white arguments are those available to the opponent. Here we see that, for example, asserting the argument b_3 would be beneficial to the proponent if the opponent knows only b_0 and b_2, while asserting b_3 would prevent the proponent from achieving its goal if the opponent knows only c_0 and c_2. The examples seen in Figures 3 and 4 illustrate the challenges posed by cycle and ladder frameworks that are not present in bipartite graphs, where asserting an argument from the partition that does not contain a goal argument can never hinder achievement of that goal. We chose to examine these argumentation framework families in order to explore the performance of our approach in particularly challenging settings.

We ran our experiments on a machine with a 2.8GHz processor, limited to 32Gb of memory. Our problems and the implementations of our translation and naive algorithm are available at: `https://goo.gl/IlcPc3`.

Time to find optimal simple strategy. For each example argumentation framework, we compared the search times taken to find an optimal simple strategy by the planner and by the naive algorithm, for opponent models of size $|\mathcal{E}| = 1, 2, 4, 8$. To account for variations in performance across opponent models of the same size, we ran our experiments on 100 randomly selected opponent models of each size with uniformly distributed probabilities (so each possible opponent model is equally likely) and empty closure operators. The average search times are shown in Table 5, with the fastest time for each setting in bold. Both approaches take < 1 second to solve almost all of the examples taken from [18] (which are, except for Dv., bipartite and each have < 10 arguments); the exception is Ex 1, which the naive algorithm finds more difficult. The planner is generally faster than the naive algorithm, but to see a significant difference we must consider the more challenging cycle and ladder examples. For the smaller instances of the cycle examples, the naive approach is faster than the planner, however for the largest instance $cycle_6$ (which comprises 13 arguments) the planner is significantly faster. The planner outperforms the naive search in all ladder examples, doing so by several orders of magnitude for $ladder_6$, which also comprises 13 arguments. Increasing improvement with problem size indicates much greater scalability of the planning approach.

Effectiveness of simple strategies. Table 5 also shows the average probability of guaranteed success of the optimal simple strategies found for each example, against the 100 randomly selected opponent models of size $|\mathcal{E}| = 1, 2, 4, 8$, with uniform probabilities and no closure. Note that λ captures the probability of *guaranteed* success, no matter what strategy the opponent employs, and thus gives a lower-bound on likelihood of success, since a simple strategy may still succeed against a possible opponent model against which it cannot guarantee success.

Since we find *optimal* simple strategies, λ is determined by the problem, not our approach. We see that size of the problem and of the opponent model has little effect on the effectiveness of simple strategies, whose probability of success depends more on the structure of the underlying argumentation framework. For 5 of the 11 examples, our optimal simple strategies give a good probability (> 0.7) of guaranteed success, while the lowest probabilities are seen for the cycle examples. We cannot know in general how much better a policy (such as those generated by the approach of Hadoux *et al.* [18]) might perform, however for the bipartite examples from [18] we can see from inspection of the problems that it is not possible to outperform an optimal simple strategy.

Opponent models with complete uncertainty. To test the limits of our approach, we also considered the case where there is no knowledge about the likelihood of the possible opponent models (and so the proponent believes all possible opponent models are equally likely, *i.e.*, its opponent model contains every element of the powerset of the arguments that are available to the opponent, each assigned the same probability). For this, we ran a single experiment with each of the examples $cycle_n$ and $ladder_n$, for $n = 4, 5, 6$, with the (unique) opponent model of size $|\mathcal{E}| = 2^k$, where k is the number of distinct arguments that could appear in an opponent model. (Note that since both the planner and the naive algorithm are deterministic it was not necessary to run multiple experiments for this setting.)

The results (Table 6) reveal that in some cases the problem remains solvable, even in the absence of any information about the opponent. Moreover, it is sometimes possible to find simple strategies that achieve a reasonable probability of success, even with complete uncertainty about the opponent's arguments. This is, once again, contingent on the underlying argumentation framework.

Example	No. args $\lvert\mathcal{A}\rvert$	Size of model $\lvert\mathcal{E}\rvert$	Planner time	Naive time	Probability of success
$cycle_4$	9	16	5.2	1.8	0.313
$cycle_5$	11	32	63.8	244.3	0.250
$cycle_6$	13	64	🕒	🕒	>0.203
$ladder_4$	9	16	9.6	6.1	0.750
$ladder_5$	11	32	457.0	107.3	0.688
$ladder_6$	13	64	🕒	🕒	>0.688

Table 6: Search times (secs.) to find an optimal simple strategy and probability of guaranteed success, with opponent models of size $|\mathcal{E}| = 2^k$, where k is the number of arguments that can appear in the opponent model. (🕒 denotes time > 1 hour.)

Example	*no closure*		$\lvert\lvert\mu\rvert\rvert = 1$		$\lvert\lvert\mu\rvert\rvert = 2$	
	Planner	Naive	Planner	Naive	Planner	Naive
$cycle_4$	0.61	0.45	1.96	0.49	2.49	0.47
(probability)	(0.41)		(0.36)		(0.24)	
$cycle_5$	19.19	8.02	36.79	8.32	51.48	8.05
(probability)	(0.30)		(0.23)		(0.17)	
$ladder_4$	0.9	2.54	1.26	2.67	1.71	2.44
(probability)	(0.74)		(0.65)		(0.57)	
$ladder_5$	12.53	50.35	15.13	42.12	23.39	34.93
(probability)	(0.66)		(0.61)		(0.47)	

Table 7: Average search times (secs.) to find an optimal simple strategy and average probability of guaranteed success, with opponent models of size $|\mathcal{E}| = 4$, without closure and with closure operators of size $||\mu|| = 1, 2$.

With examples $cycle_6$ and $ladder_6$, both the planner and the naive search took longer than one hour to complete their searches and were terminated prematurely. Consequently we are unable to report the optimal probability of success that can be obtained for these examples. However from the naive search we are able to place a lower bound on the optimal success rate by reporting the highest probability reported after one hour.

Effect of closure. We may measure the 'size' or sophistication of a closure operator μ by the maximal number of new arguments it is able to deduce from any given set, *i.e.*, $||\mu|| = \max_{\mathcal{B}\subseteq\mathcal{A}} |\mu(\mathcal{B}) - \mathcal{B}|$. To investigate the effect the closure operator has on the performance of both the planner and the naive search, for each of the examples $cycle_n$ and $ladder_n$, for $n = 4, 5$, we selected a random sample of 100 opponent models of size $|\mathcal{E}| = 4$, where each possible opponent model is equally likely and whose closure operators all have size $||\mu|| = k$, for $k = 0, 1, 2$. We report the average times taken by both the approaches to find an optimal simple strategy, together with the average probability of guaranteed success of the optimal simple strategies found (Table 7).

We found that the addition of a closure operator has negligible effect on the speed of the naive search but a modest negative effect on the performance of the planner; *e.g.*, for $cycle_5$ (resp. $ladder_5$), the addition of a closure operator of size 2 increased the average planner time from 19.19 to 51.48 secs. (resp. 12.53 to 23.39 secs.). There is also a steady decrease in the probability of success of our simple strategies as we allow our opponent to employ more sophisticated reasoning, with the addition of a closure operator of size 2 decreasing the average probability of guaranteed success by ≈ 0.2.

6. DISCUSSION

We believe our approach is the first to use an automated planner to generate proponent strategies for symmetric persuasion that, with a certain probability, guarantee success no matter which arguments the opponent asserts. Furthermore, our approach accounts for the fact that the opponent may use knowledge from arguments asserted by the proponent to construct new arguments not known to the opponent at the start of the dialogue (using the closure operator). It is important to account for this when strategising, as the opponent may be able to use such arguments against the proponent. As the works of Rienstra *et al.* [36] and Rosenfeld and Kraus [39] treat arguments as abstract entities and do not provide a mechanism for inferring new arguments, these cannot account for such phenomena. While Hadoux *et al.* [18] and Hadoux and Hunter [20] do not explicitly consider that the opponent may be able to infer new arguments from knowledge gained from the opponent, the representations they use to model the strategic argumentation problems they address (respectively, executable logic for dialogical argumentation [6], and a mass distribution that captures the opponent's belief in the available arguments) seem rich enough to capture such cases; it is unclear, however, what effect this would have on the performance of these approaches.

Like our approach, those of Hadoux *et al.* [18], Hadoux and Hunter [20] and Ronsenfeld and Kraus [39] assume that the proponent is aware of all the arguments that may be available to the opponent. The approach of Rienstra *et al.* [36] allows the proponent to consider arguments that it itself is not aware of but believes are known to the opponent (referred to as *virtual arguments*); however this requires that the proponent is aware of exactly the attack relationships between each virtual and all other arguments, which can be captured equivalently in our model by an argument that appears in the proponent's model of the opponent, but is not part of the arguments available to the proponent. It may occur during a dialogue that the opponent asserts an argument that is not part of the proponent's opponent model (whatever form this takes) in which case the effectiveness of the generated strategy is no longer assured. We plan in future work to investigate how, in such a case, we might update our opponent model (as is considered in, *e.g.*, [7, 17, 19, 22, 24, 25, 26, 36]) in order to replan our simple strategy.

We have shown we can deal with challenging examples with up to 13 arguments, where cycles in the underlying argumentation framework mean that an argument may be either helpful or harmful for the proponent, depending on the arguments actually available to the opponent and the opponent's strategy. Our results show that, despite our restriction to simple strategies, we can guarantee reasonable chance of success, regardless of how the opponent plays. While other approaches [18, 20, 39] generate richer policies than our simple strategies, their performance on examples involving cycles in the underlying argumentation framework is yet to be explored. Scalability is still an issue for our approach, due to the PSPACE-hardness of the problem [29], however a study of 19 debates from Debatepedia found an average of 11 arguments per debate [9], suggesting we can deal with realistically sized dialogues. In future, we aim to improve both scalability and effectiveness of our approach by solving for partitions of the opponent model and investigating ways to combine the resulting simple strategies to create a more effective policy. We also plan to adapt our planning model to take account of information about the expected behaviour of the opponent when this is available, and to explore further the relationship between the problem structure and the effectiveness of an optimal simple strategy.

Acknowledgements

This research was partly funded by the EPSRC, grant reference EP/M01892X/1, for the Planning an Argument project. We thank the anonymous reviewers of this work for their valuable comments.

REFERENCES

[1] A. Albore, H. Palacios, and H. Geffner. Compiling uncertainty away in non-deterministic conformant planning. In *Proceedings of the 19th European Conference on Artificial Intelligence*, pages 465–470, 2010.

[2] T. J. Bench-Capon and P. E. Dunne. Argumentation in artificial intelligence. *Artificial intelligence*, 171(10-15):619–641, 2007.

[3] E. Black and K. Bentley. An empirical study of a deliberation dialogue system. In *Proceedings of the 1st International Workshop on the Theory and Applications of Formal Argumentation*, pages 132–146, 2012.

[4] E. Black, A. J. Coles, and S. Bernardini. Automated planning of simple persuasion dialogues. In *Proceeings of the 15th International Workshop on Computational Logic in Multi-Agent Systems*, pages 87–104, 2014.

[5] E. Black, A. J. Coles, and C. Hampson. Optimal simple strategies for persuasion. In *Proceedings of the 22nd European Conference on Artificial Intelligence*, pages 1736–1737, 2016.

[6] E. Black and A. Hunter. Executable logic for dialogical argumentation. In *Proceedings of the 20th European Conference on Artificial Intelligence*, pages 15–20, 2012.

[7] E. Black and A. Hunter. Reasons and options for updating an opponent model in persuasion dialogues. In *Proceedings of the 3rd International Workshop on the Theory and Applications of Formal Argumentation*, pages 21–39, 2015.

[8] T. Bylander. The computational complexity of propositional STRIPS planning. *Artificial Intelligence*, 69(1-2):165–204, 1994.

[9] E. Cabrio and S. Villata. A natural language bipolar argumentation approach to support users in online debate interactions. *Argument & Computation*, 4(3):209–230, 2013.

[10] M. Caminada. A discussion game for grounded semantics. In *Proceedings of the 3rd International Workshop on Theory and Applications of Formal Argumentation*, pages 59–73, 2015.

[11] D. Carmel and S. Markovitch. Learning and using opponent models in adversary search. *Technical Report, CIS9606, Technion*, 1996.

[12] G. Charwat, W. Dvořák, S. A. Gaggl, J. P. Wallner, and S. Woltran. Methods for solving reasoning problems in abstract argumentation: A survey. *Artificial Intelligence*, 220:28 – 63, 2015.

[13] A. J. Coles, A. I. Coles, M. Fox, and D. Long. Forward-chaining partial-order planning. In *Proceedings of the 20th International Conference on Automated Planning and Scheduling*, pages 42–49, 2010.

[14] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77(2):321 – 357, 1995.

[15] P. E. Dunne and M. Wooldridge. Complexity of abstract argumentation. In G. Simari and I. Rahwan, editors, *Argumentation in Artificial Intelligence*, pages 85–104. Springer US, 2009.

[16] M. Fox and D. Long. PDDL2. 1: An extension to PDDL for expressing temporal planning domains. *Journal of Artificial Intelligence Research*, 20:61–124, 2003.

[17] C. Hadjinikolis, S. Modgil, E. Black, P. McBurney, and Y. Siantos. Opponent modelling in persuasion dialogues. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, pages 164–170, 2013.

[18] E. Hadoux, A. Beynier, N. Maudet, P. Weng, and A. Hunter. Optimization of probabilistic argumentation with markov decision models. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, pages 2004–2010, 2015.

[19] E. Hadoux and A. Hunter. Computationally viable handling of beliefs in arguments for persuasion. In *Proceedings of the 28th International Conference on Tools with Artificial Intelligence*, pages 319–326, 2016.

[20] E. Hadoux and A. Hunter. Strategic sequences of arguments for persuasion using decision trees. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, 2017. (In press).

[21] J. Hoffmann. The Metric-FF Planning System: Translating "Ignoring Delete Lists" to Numeric State Variables. *Journal of Artificial Intelligence Research*, 20:291–341, 2003.

[22] A. Hunter. Modelling the persuadee in asymmetric argumentation dialogues for persuasion. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, pages 3055–3061, 2015.

[23] A. Hunter. Computational persuasion with applications in behaviour change. In *Proceedings of the 6th International Conference on Computational Models of Argument*, pages 5–18, 2016.

[24] A. Hunter. Persuasion dialogues via restricted interfaces using probabilistic argumentation. In *Proceedings of the 10th International Conference on Scalable Uncertainty Management*, pages 184–198, 2016.

[25] A. Hunter. Two dimensional uncertainty in persuadee modelling in argumentation. In *Proceedings of the 22nd European Conference on Artificial Intelligence*, pages 150–157, 2016.

[26] A. Hunter and M. Thimm. Optimization of dialectical outcomes in dialogical argumentation. *International Journal of Approximate Reasoning*, 78:73–101, 2016.

[27] M. Kacprzak, M. Dziubinski, and K. Budzynska. Strategies in dialogues: A game-theoretic approach. In *Proceedings of the 5th International Conference on Computational Models of Argument*, pages 333–344, 2014.

[28] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1):99–134, 1998.

[29] M. J. Maher. Complexity of exploiting privacy violations in strategic argumentation. In *Proceedings of the 13th Pacific Rim International Conference on Artificial Intelligence*, pages 523–535, 2014.

[30] P.-A. Matt and F. Toni. A game-theoretic measure of argument strength for abstract argumentation. In *Proceedings of the 11th European Conference on Logics in Artificial Intelligence*, pages 285–297, 2008.

[31] S. Modgil, F. Toni, F. Bex, I. Bratko, C. I. Chesñevar, W. Dvořák, M. A. Falappa, X. Fan, S. A. Gaggl, A. J. García, M. P. González, T. F. Gordon, J. Leite, M. Možina, C. Reed, G. R. Simari, S. Szeider, P. Torroni, and S. Woltran. The added value of argumentation. In S. Ossowski, editor, *Agreement Technologies*, pages 357–403. Springer Netherlands, 2013.

[32] S. C. Ong, S. W. Png, D. Hsu, and W. S. Lee. Planning under uncertainty for robotic tasks with mixed observability. *The International Journal of Robotics Research*,

29(8):1053–1068, 2010.

[33] E. P. D. Pednault. ADL: Exploring the middle ground between STRIPS and the situation calculus. In *Proceedings of the 1st International Conference on Principles of Knowledge Representation and Reasoning*, pages 324–332, 1989.

[34] H. Prakken. Formal systems for persuasion dialogue. *The Knowledge Engineering Review*, 21(02):163–188, 2006.

[35] I. Rahwan and K. Larson. Mechanism design for abstract argumentation. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1031–1038, 2008.

[36] T. Rienstra, M. Thimm, and N. Oren. Opponent models with uncertainty for strategic argumentation. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, pages 332–338, 2013.

[37] R. Riveret, H. Prakken, A. Rotolo, and G. Sartor. Heuristics in argumentation: A game-theoretical investigation. In *Proceedings of the 2nd International Conference on Computational Models of Argument*, pages 324–335, 2008.

[38] A. Rosenfeld and S. Kraus. Providing arguments in discussions on the basis of the prediction of human argumentative behavior. *ACM Transactions on Interactive Intelligent Systems*, 6(4):30:1–30:33, 2016.

[39] A. Rosenfeld and S. Kraus. Strategical argumentative agent for human persuasion. In *Proceedings of the 22nd European Conference on Artificial Intelligence*, pages 320–328, 2016.

[40] K. Satoh and K. Takahashi. A semantics of argumentation under incomplete information. In *International Workshop on Juris-informatics*, 2011.

[41] M. Thimm. Strategic argumentation in multi-agent systems. *Künstliche Intelligenz, Special Issue on Multi-Agent Decision Making*, 28(3):159–168, 2014.

[42] M. Thimm, S. Villata, F. Cerutti, N. Oren, H. Strass, and M. Vallati. Summary report of the first international competition on computational models of argumentation. *AI Magazine*, 37(1):102–104, April 2016.